Evaluation of Machine Learning Methods for Natural Language Processing Tasks

Walter Daelemans, Véronique Hoste

CNTS Language Technology Group, University of Antwerp
UIA, Universiteitsplein 1 (bldng A), B-2610 Antwerpen, Belgium
{daelem,hoste}@uia.ua.ac.be

Abstract

We show that the methodology currently in use for comparing symbolic supervised learning methods applied to human language technology tasks is unreliable. We show that the interaction between algorithm parameter settings and feature selection within a single algorithm often accounts for a higher variation in results than differences between different algorithms or information sources. We illustrate this with experiments on a number of linguistic datasets. The consequences of this phenomenon are far-reaching, and we discuss possible solutions to this methodological problem.

1. Introduction

Symbolic supervised machine learning methods (e.g. Inductive Logic Programming (ILP), Decision Tree Induction, Rule Induction, Memory-Based Learning, etc.) have recently taken front stage in language technology. Most natural language processing (NLP) problems can be formulated as classification problems (given some object and its context, decide on the class of this object). Typical instances of this type of problem are part-of-speech tagging and word sense disambiguation. Supervised learning methods work by extracting regularities from a set of examples (e.g. collected from an annotated corpus). The reason why these methods are researched intensively is that, like statistical approaches, they are often reported to achieve higher efficiency, more robustness, and better coverage than handcrafting approaches. On top of this, they are reported to have a number of advantages compared to statistical approaches. E.g., ILP systems allow easy integration of linguistic background knowledge in the learning system, induced rule systems are interpretable, memory-based learning methods incorporate smoothing by similarity-based learning, etc.

Supervised symbolic machine learning methods differ in their *bias*, i.e. the built-in constraints they have in what can be represented as an induced hypothesis, and how the search for a hypothesis is heuristically guided[1]. Some examples of bias are the fact that decision tree learners favor compact decision trees, and that ILP systems can represent hypotheses in terms of first order logic in contrast to most other learning methods which can only represent propositional hypotheses. Given these widely differing biases, it is important to study which algorithm "has the right bias" for learning language.

Theoretical studies in Machine Learning (e.g. the *no free lunch* theorem (Wolpert and Macready, 1995)) have shown that no inductive algorithm is universally better than any other: generalization performance of any inductive algorithm is zero when averaged over a uniform distribution of all possible classification problems. In order to know which learning algorithm has the right bias for language learning, it is therefore necessary to compare machine learning methods experimentally on their behavior on some specific, typical, language processing task (e.g. POS tagging, discourse segmentation, etc.). A posteriori, we may be able to say something about the bias of a particular class of algorithms being suited or not for a particular class of problems. E.g., in (Daelemans et al., 1999) it is claimed that memory-based learning has the right bias for NLP disambiguation in context tasks because it does not abstract from exceptions and low-frequency events.

In NLP, this comparative approach has gained an enormous importance with the influence of competitive research evaluatons such as MUC, DUC, SENSEVAL, and the CoNLL shared tasks. Although in many cases also handcrafted and complex systems are entered in such a competition, a large part of this research is based on the comparison of machine learning algorithms for some subtask, the type of experiment we are targeting in this paper.

In summary, for lack of a priori knowledge about which algorithm is suited for which task, the goal of a large part of current research in Machine Learning of Natural Language is to investigate which type of learning algorithm has the right *bias* for typical Natural Language Processing tasks (e.g. disambiguation in context). The approach used to achieve this are *comparative machine learning experiments*, for which a detailed experimental methodology has been developed. It is to this methodology and its limitations that we now turn.

2. Comparative Machine Learning Methodology

Crucial for objectively comparing algorithm bias and information source contribution is a methodology to reliably measure differences and compute their statistical significance. A detailed methodology has been developed for this (Weiss and Kulikowski, 1991; Weiss and Indurkhya, 1998) involving approaches like k-fold cross-validation (Kohavi and John, 1997; Alpaydin, 1999; Dietterich, 1998) to estimate classifier error (or derived measures like precision, recall, and F-score), and statistical techniques like McNemar (Dietterich, 1998) and paired cross-validation t-

[1] The same applies to unsupervised learning methods, but we will not discuss these further in this paper.

tests for determining the statistical significance of differences between algorithms or between presence or absence of information sources. Although this methodology is not without its problems (Salzberg, 1997), it is generally accepted and used both in Machine Learning and in most work in statistical NLP. A seminal paper on the comparison of the accuracy of different machine learning (ML) methods (Mooney, 1996) on the task of word sense disambiguation (WSD) is a good example of best practice in this approach. Many more examples can be found in the recent NLP literature of similar studies and interpretations. He tested seven ML algorithms on their ability to disambiguate the word *line*, and concluded that within the class of symbolic machine learning methods, decision lists are at an advantage for WSD because of their rule ordering bias. Analogous to decision trees and much like rule induction approaches, decision lists search for a minimal-size ordered set of high-accuracy rules, that disambiguate efficiently and effectively. This is as interesting and useful result, but we will show that interpretations like these are not necessarily reliable.

As an informal characterization of a comparative machine learning experiment, we isolate the following components. The data set used (before cross-validation) is the *sample selection*, the selection of information sources used as input to the ML algorithm (the features chosen as predictors for the output class) is the *feature selection*, these features are presented in some *feature representation* (e.g. binary, numeric, nominal), and the algorithm(s) chosen have a particular *algorithm parameter setting*[2]. In a typical, methodologically correct, comparative machine learning experiment, two algorithms (A and B) are compared for a fixed sample selection, feature selection, and feature representation over a number of trials (cross-validation), and if the measured differences are statistically significant, conclusions are drawn about which algorithm is better suited and why (mostly in terms of algorithm bias). Most of the time, the algorithm parameter settings are kept fixed as well (mostly using the default settings), although sometimes the parameters are optimized on the training data.

Our hypothesis is that the accuracy differences which can be observed between different machine learning algorithms on some problem using standard methodology will in general be lower than the variability in accuracy resulting from interactions between sample selection, algorithm parameter settings, and information source selection and representation. This makes many published results (and their interpretation) unreliable.

In Section 3., we describe an experiment to test that our hypothesis holds for the interaction between feature selection and algorithm parameter setting on a series of benchmark NLP tasks when comparing memory-based learning and rule induction. Section 4. shows that the effect of (possible) algorithm bias on generalization accuracy is easily overwhelmed by the effect of algorithm parameter variation, and the interaction between feature selection and algorithm parameter setting. In Section 6. we integrate our results with related research, and propose possible solutions to the problem.

[2] An additional dimension in a general model could be combination methods like bagging and boosting.

3. Experiments

To test our hypothesis, we analyzed the impact of algorithm parameter optimization, and the interaction between feature selection and algorithm parameter optimization on classifier accuracy in a comparative experiment.

Feature (subset) selection is the process in which a subset of the available predictor features defining the input of the classification task are removed if they can't be shown to be relevant in solving the learning task (Kohavi and John, 1997). Often instead of removing features, the features are assigned relevance weights (Wettschereck et al., 1997). Without computational complexity restrictions, the ideal way to do feature selection would be to try all possible feature subsets and keep the one with the highest accuracy on some holdout set. In practice, a heuristic approach is the only feasible way, E.g., we start with each feature independently as only feature, compute accuracy, then select the feature with highest accuracy and compute accuracy of pairs of features when each other feature is added, and so on, until no more accuracy increase is reported (forward selection). Alternatively we can start with all available features and look at the effect on accuracy of deleting one of the features, and continue deleting until no more accuracy increase is reported (backward selection).

Algorithm parameter optimization is the process in which parameters of a learning system (e.g. learning rate for neural networks, or the number of nearest neighbors in memory-based learning), are tuned for a particular problem. Although most machine learning systems provide sensible default settings, it is by no means certain that they will be *optimal* parameter settings for some particular task.

In both cases (feature selection and parameter optimization), we are performing a *model selection* task which is well-known in Machine Learning. Whereas some published work in computational linguistics discusses either feature selection for some task, or algorithm parameter optimization on training data, the effects of their interaction have as far as we know never been studied.

The general set-up of our experiments is the following. Each experiment is done using a 10-fold cross-validation on the available data. This means that the data is split in 10 partitions, and each of these is used once as test set, with the other nine as corresponding train set. For each dataset, we provide information about the accuracy of both ML systems:

1. Using their default settings.
2. When optimizing the algorithm parameters for each algorithm individually. Each "reasonable" parameter setting is used during training on each of the ten folds, and the best setting is used on the test set of that fold. (Optimization step 1).
3. When doing feature selection (forward) interleaved with optimization of the parameters for each algorithm. Within each step of the forward feature selection method, the parameter optimization procedure is performed for each fold. (Optimization step 2).

From our hypothesis, we expect that (i) each optimization step can increase the accuracy of the best result (as measured by the average result over the 10 experiments) considerably, and (ii) that the "best" algorithm for some task using default settings is not necessarily the best algorithm when doing the optimizations. In general, we expect the variability we record for the same algorithm over the three conditions to be much larger than the difference between the two learning algorithms. In the remainder of this Section we will describe the learning algorithms and datasets that were used.

3.1. Machine Learning Methods

We chose two machine learning techniques for our experiments, memory-based learning (TIMBL) and decision-tree learning (Ripper).

The distinguishing feature of memory-based learning (MBL) in contrast with "eager" ML algorithms (e.g. decision trees) is that MBL keeps all training data in memory, and only abstracts at classification time by extrapolating a class from the most similar item(s) in memory to the new test item. This strategy is often referred to as "lazy" learning. For our experiments, we used the MBL algorithms implemented in TIMBL[3]. We give a brief overview of the algorithms and metrics here, and refer to (Daelemans et al., 1997; Daelemans et al., 2001) for more information.

IB1: The distance between a test item and each memory item is defined as the number of features for which they have a different value. Unlike in (1991), here classification occurs via the *k-nearest-distances* rule: all memory items which are equally near at the nearest k distances surrounding the test item are taken into account in classification. The classification assigned to the test item is simply the majority class among the memory items at the k nearest distances.

IGTREE: An oblivious decision tree is created with features as tests, and ordered according to one of the feature weighting methods discussed earlier, as a heuristic approximation of the computationally more expensive pure k-nearest distance classifier.

A good choice of parameter settings can have a large effect on accuracy in TIMBL. In TIMBL, the following algorithm parameters could be optimized:

- *Optimization of the similarity metrics*: overlap (default; the distance between two patters is the sum of the differences between the features) and Modified Value Difference Metric (MVDM, the similarity of the values of a feature are determined by looking at co-occurrence of values with target classes).
- *Optimization of different feature weighting metrics*: no weighting, gain ratio weighting (default), information gain weighting, chi-squared weighting and shared variance weighting (see (Daelemans et al., 2001)).
- *Optimization of the class voting weights* that are used for extrapolation from the nearest neighbour set: normal majority voting (default), inverse distance weighting, inverse linear weighting, exponential decay weighting.
- *Optimization of the k value*, representing the number of nearest distances in which memory items are searched. In the experiments, k was varied between 1 (default), 3, 5, 7, 9, 11, 15, 25, 35 and 45.

As a second learning algorithm, we used RIPPER (Cohen, 1995) a member of the family of rule learning algorithms. RIPPER's hypothesis is expressed as a set of if-then rules. Before learning, RIPPER first heuristically orders the classes. After arranging the classes, it finds rules to distinguish between the classes. The final class becomes the default class.

For RIPPER the following parameter settings were varied:

- *Optimization of the class ordering*: order by increasing frequency (default), order by decreasing frequency, miminal-description-length (use MDL heuristic to guess an optimal ordering).
- *Allow (default) or disallow negative tests in the rule conditions.*
- *Optimization of hypothesis simplification*, in which the values < 0 denote less simplification and the values $>$ more simplification. We set the values to 0.5 (default), 1 and 1.5.
- *Optimization of example coverage*, in which the rules are forced to cover at least # examples. We varied the number of examples between 1, 2 (default), 3 and 4.

3.2. Data

Three types of natural language data sets were used, involving the prediction of word senses (5 data sets), diminutive suffixes (1 data set) and part-of-speech categories (2 data sets).

- The **word sense disambiguation** data sets are extracted from the Semcor corpus included in WordNet1.6. In this corpus, every word is linked to its appropriate sense in the WordNet sense lexicon. From this corpus, five ambiguous words were selected: "line" (124 instances), "little" (205 instances), "make" (752 instances), "then" (479 instances) and "time" (509 instances). The task of word sense disambiguation consists in predicting the correct sense for an ambiguous word in a given context. The input vector for the experiments represents the local context of the focus word in a window of three word forms to the left and three to the right. For the focus word, both the lemma and POS are provided. For the context words forms, POS information is given. E.g. the following is a training instance for the word "little": `came VBD from IN a DT little little JJ Pennsylvania NNP town NN near JJ, little%3:00:01::`
- The **diminutive** data set (3949 instances) is extracted from the CELEX lexical data base (Baayen et al., 1993) and involves the prediction of the diminutive suffix form in Dutch. In Dutch, a noun can receive a

[3] Available from http://ilk.kub.nl

diminutive suffix to indicate "small size", e.g. "boompje" stands for "little tree". This diminutive suffix shows variation in its form: -tje, -etje, -pje, -kje. The task is to predict which suffix form is chosen for previously unseen words on the basis of their form. For the three last syllables of the noun, four different features are collected: (i) whether the syllable is stressed or not (+ or -), (ii) the string of consonants before the vocalic part of the syllable, (iii) its vocalic part, (iv) its post-vocalic part. The sign "=" is used whenever a feature is not present. E.g. the following is a training instance for the word "bijbel" (Eng. "bible", diminutive "bijbeltje"): `= = = = + b K = - b @ l, T`.

- The **part-of-speech** data set is based on the TOSCA tagged LOB corpus of English. There are two versions of the data, one version involved with predicting the part-of-speech for known words and one for the unknown words. The features in the known words data set represent information about the possible categories of the focus word to be tagged and its context. For the unknown words, we have to rely on the context and the wordform only. In both data sets, the feature vector represents the coded POS tags of two words before and two words after the focus word, the last three letters of the focus word and information about hyphenation and capitalization. The known words data set has two extra features: the focus word itself and its ambiguity class. E.g. the following is a training instance for the word "house" (known words data set): `house Bn AU AN AE AA u s e 0 0, AN`.

Now that we have introduced the approach we took to test our hypothesis, our data, and the learning algorithms to be compared, we can move to the results and their interpretation in the next Section.

4. Results

In Table 1, we show that it is indeed incorrect to conclude that one algorithm is better than another for a certain classification problem merely on the basis of default algorithm parameter settings: for 5 data sets ("line", "little", "make", diminutive and tag known), RIPPER outperforms TIMBL when using default parameter settings. However, when optimizing the algorithm parameters, we observe that the algorithm which performed best with the standard settings, is not necessarily the best algorithm when doing the optimizations.

When looking at the variation for a single algorithm over the three conditions, we can see that parameter optimization and combined feature selection with parameter optimization leads to major accuracy improvements compared to the results obtained with default parameter settings. These 'vertical' accuracy differences are much larger than the 'horizontal' algorithm-comparing accuracy differences. The fact that we could observe large standard deviations in the optimization experiments, also confirms the necessity of parameter optimization (only the best result is represented in Table1 for each optimization step).

		TIMBL	Ripper
"line"	(i)	20.19	**21.77**
	(ii)	**27.31**	22.58
	(iii)	**38.59**	33.87
"little"	(i)	62.81	**73.17**
	(ii)	**76.55**	74.63
	(iii)	**80.50**	79.02
"make"	(i)	42.17	**46.94**
	(ii)	**54.13**	49.47
	(iii)	**58.25**	53.19
"then"	(i)	**68.47**	67.43
	(ii)	**73.70**	69.31
	(iii)	**77.89**	72.65
"time"	(i)	**56.39**	47.94
	(ii)	**62.28**	58.74
	(iii)	**64.83**	62.28
Diminutive	(i)	96.02	**96.25**
	(ii)	**97.82**	97.34
	(iii)	**97.87**	97.57
Tag unknown	(i)	**76.3**	76.09
	(ii)	**82.22**	78.06
	(iii)	**82.22**	78.06
Tag known	(i)	93.02	**93.11**
	(ii)	**95.23**	93.91
	(iii)	**96.46**	94.45

Table 1: Results of TIMBL and RIPPER on the different natural language data sets when using (i) default settings, (ii) parameter optimization and (iii) forward selection with parameter optimization.

With respect to the selected parameter settings and feature combinations, we found that those parameter settings which are selected after optimization cannot be generalized over the different data sets, not even within a single data set (parameter settings which are optimal when using all features are not necessarily optimal when performing feature selection). Furthermore, we could observe for all data sets that the feature selection considered to be optimal for TIMBL could be different from the one optimal for RIPPER.

We conclude that we can confirm our hypothesis that the accuracy differences between different machine learning algorithms using standard comparative methodology will in general be lower than the variability in accuracy resulting from interactions between algorithm parameter settings and information source selection.

4.1. Results on a hold-out test set

The best results in each of the optimization steps of Table 1 are no guarantee that these accuracies will also be achieved with independent test material. Optimization happened on each test fold in the 10-fold cross-validation (rather than using an additional cross-validation), which is methodologically acceptable as we were only interested in the variability of the results.

For studying the effect on the real error, we therefore tested our approach also on a hold-out test set. For both tagging data sets, we took an independent test set of 5000 instances coming from the same LOB corpus described in Section 4. We performed the experiments with the parameter settings/feature subsets which were considered to be

		TIMBL	Ripper
Tag unkn.	(i)	**76.56**	76.18
	(ii)	**81.60**	78.48
	(iii)	**81.60**	78.48
Tag kn.	(i)	92.54	**93.02**
	(ii)	**94.56**	93.28
	(iii)	**95.46**	93.2

Table 2: Results of TIMBL and Ripper on the tagging data sets when using (i) default settings, (ii) the optimal parameter settings, and (iii) the optimal feature subset with the optimal parameter settings.

optimal after cross-validation on the train set. The results of these experiments are displayed in Table 2, and confirm the results of the validation experiments (Table 1).

5. Solutions

Optimization problems of the type described in this paper are of course not unique to machine learning of language. Solutions like *genetic algorithms* (GAs) have been proposed as domain-independent techniques suitable for exploring large search spaces such as those described in this paper. GAs start with a collection of randomly initialized hypothesis solutions in the form of symbolic strings called chromosomes or individuals. The chromosome consists of a series of genes each of which represents a particular feature-value combination. This population is evolved by selecting the fittest individuals and recombining them by means of mutation and crossover operators. Features can be easily represented as binary "genes" that can be on or off. A particular feature subset selection is then represented as those feature genes that are on. Different parameter settings of the learning algorithm can also be represented as genes (e.g., the value of k as a numeric gene with integer values). By combining these representations, we can make the GA search for an optimal combination of parameter settings and feature selection. The fitness measure for each individual is based on the accuracy of the classifier corresponding to its chromosome on a test set (or by cross-validation). This approach can be extended to other aspects of the learning situation such as feature representation and sample selection (at the cost of more computation).

In previous research, we have applied genetic algorithms to feature selection and combined feature selection and algorithm parameter setting. However, we found that GA feature selection does not significantly do better than the simple feature selection methods described above (Kool et al., 2000a), and that whereas GA combined parameter optimization and feature selection is feasible it does not always show performance gains (Kool et al., 2000b). These are prliminary results, however, merely scratching the surface of possible experiments, and further research is needed.

6. Related Research and Conclusion

An argument similar to ours can be found in (Banko and Brill, 2001) for the effect of one aspect of sample selection (training set size). For the task of disambiguating *confusables* (words like *it's* and *its* which are easily confused in writing), increasing training data with a factor 10^3 has a significantly larger effect on generalization accuracy than the choice of algorithm on the "smaller" (still 1 million cases) training set, and the effect of algorithm bias becomes considerably smaller at these large training set sizes.

Our research shows that at the 1 million end of their learning curve, the isolated points (representing different machine learning method results) should actually be interpreted as largely overlapping intervals. And that even there, comparative results are not reliable. It would be interesting to investigate whether the variability due to the interaction between feature selection and algorithm parameter optimization decreases as the training set size increases. If this were not the case, then even at large training set sizes, we cannot be sure of differences found with default settings. Our computational resources didn't allow us to investigate this.

In (Banko and Brill, 2001) it is concluded that "We have no reason to believe that any comparative conclusions drawn on one million words will hold when we finally scale up to larger training corpora", and they therefore advise a switch of attention in the field from comparative experiments on "small" (1 million) datasets to techniques for efficiently collecting more training data. From our experiments we could add that there is also no reason to believe that any comparative conclusions drawn on the basis of standard comparative machine learning methodology will hold when sufficient optimalization of algorithm parameters and feature selection is available. More attention should therefore also be given to efficient optimalization stratiegies where computationally feasible.

7. References

D.W. Aha, D. Kibler, and M. Albert. 1991. Instance-based learning algorithms. *Machine Learning*, 6:37–66.

Ethem Alpaydin. 1999. Combined 5×2 cv F test for comparing supervised classification learning algorithms. *Neural Computation*, 11(8):1885–1892.

R.H. Baayen, R. Piepenbrock, and H. van Rijn. 1993. *The CELEX lexical data base on CD-ROM*. Philadelphia: Linguistic Data Consortium.

M. Banko and E. Brill. 2001. Scaling to very very large corpora for natural language disambiguation. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*, pages 26–33. Association for Computational Linguistics.

W.W. Cohen. 1995. Fast effective rule induction. In *Proc. 12th International Conference on Machine Learning*, pages 115–123. Morgan Kaufmann.

W. Daelemans, A. van den Bosch, and T. Weijters. 1997. Igtree: Using trees for compression and classification in lazy learning algorithms. *Artificial Intelligence Review, special issue on Lazy Learning*, 11:407–423.

Walter Daelemans, Antal van den Bosch, and Jakub Zavrel. 1999. Forgetting exceptions is harmful in language learning. *Machine Learning*, 34:11–41.

W. Daelemans, J. Zavrel, K. van der Sloot, and A. van den Bosch. 2001. Timbl: Tilburg memory based learner,

version 4.0, reference guide. Technical report, ILK Technical Report 01-04.

Thomas G. Dietterich. 1998. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, 10(7):1895–1923.

Ron Kohavi and George H. John. 1997. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1–2):273–323.

Anne Kool, Walter Daelemans, and Jakub Zavrel. 2000a. Genetic algorithms for feature relevance assignment in memory-based language processing. In Claire Cardie, Walter Daelemans, Claire Nédellec, and Erik Tjong Kim Sang, editors, *Proceedings of the Fourth Conference on Computational Natural Language Learning and of the Second Learning Language in Logic Workshop, Lisbon, 2000*, pages 103–106. Association for Computational Linguistics, Somerset, New Jersey.

Anne Kool, Jakub Zavrel, and Walter Daelemans. 2000b. Simultaneous feature selection and parameter optimization for memory-based natural language processing. In Ad Feelders, editor, *Proceedings of the 10th BENELEARN meeting*, pages 93–100. Tilburg, The Netherlands.

Raymond J. Mooney. 1996. Comparative experiments on disambiguating word senses: An illustration of the role of bias in machine learning. In Eric Brill and Kenneth Church, editors, *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 82–91. Association for Computational Linguistics, Somerset, New Jersey.

Steven L. Salzberg. 1997. On comparing classifiers: Pitfalls to avoid and a recommended approach. *Data Mining and Knowledge Discovery*, 1(3):317–327.

Sholom Weiss and Nitin Indurkhya. 1998. *Predictive Data Mining: A Practical Guide*. Morgan Kaufmann, San Francisco.

Sholom M. Weiss and Casimir A. Kulikowski. 1991. *Computer Systems That Learn: Classification and Prediction Methods from Statistics, Neural Nets, Machine Learning, and Expert Systems*. Morgan Kaufmann, San Mateo, California.

Dietrich Wettschereck, David W. Aha, and Takao Mohri. 1997. A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms. *Artificial Intelligence Review*, 11(1-5):273–314.

David H. Wolpert and William G. Macready. 1995. No free lunch theorems for search. Technical Report SFI-TR-95-02-010, Santa Fe Institute, Santa Fe, NM.